Journal of Global Optimization **11:** 163–180, 1997.
© 1997 *Kluwer Academic Publishers. Printed in the Netherlands.*

A Pseudo-Polynomial Primal-Dual Algorithm for Globally Solving a Production-Transportation Problem

TAKAHITO KUNO* and TAKAHIRO UTSUNOMIYA
Institute of Information Sciences and Electronics, University of Tsukuba, Tsukuba, Ibaraki 305, Japan

(Received: 14 January 1997; accepted: 28 January 1997)

Abstract. In this paper, we propose a primal-dual algorithm for solving a class of production-transportation problems. Among $m(\geq 2)$ sources two factories exist, which produce given goods at some concave cost and supply them to n terminals. We show that one can globally minimize the total cost of production and transportation by solving a Hitchcock transportation problem with m sources and n terminals and a minimum linear-cost flow problem with $m + n$ nodes. The number of arithmetic operations required by the algorithm is pseudo-polynomial in the problem input length.

Key words: Global optimization, production-transportation problem, minimum concave-cost flow problem, primal-dual algorithm, pseudo-polynomial algorithm.

1. Introduction

Suppose a firm has m sources of given goods, p of which are factories and the rest are warehouses. There are n terminal stores dealing in the goods. The decision maker of the firm has to cope with the demands of these terminals, so as to minimize the total cost of producing the goods and of shipping them to each terminal. This is the production-transportation problem which we consider in the paper.

Due to economies of scale, the production cost is in general a nondecreasing and concave function of the output. This means that the production-transportation problem has multiple locally optimal solutions, many of which fail to be globally optimal. Hence, the problem belongs to a class of multiextremal global optimization [10]. Although such a problem is usually difficult to solve, a number of promising algorithms are proposed for some network problems (see [8,7] for the current state-of-the-art of nonconvex network optimization).

In their recent articles [19, 20], Tuy *et al.* proposed a strongly polynomial algorithm for solving a special type of production-transportation problems, where the number p of factories is fixed at two or three and warehouses are absent, i.e. $m = p$. Their result sharply contrasts with general concave minimization problems, which are NP-hard even when just one variable is nonlinear [17]. They have further devel-

* The author was partially supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture, Grant No. (C)07680447.

oped this algorithm to solve the problem with any fixed $m = p$ in a subsequent article [21]. Another special type has been studied in our article [13], where warehouses are absent again but the number p of factories is not fixed. We assumed that terminals are partitioned into $p-1$ disjoint sets and each of $p-1$ factories is allowed to supply only its assigned set of terminals. We exploited this network structure and solved the problem in time $O(Bnp)$, where B represents the total demand of terminals.

In this paper, we assume that $m \geq p$, i.e. there can be some warehouses, each of which produces nothing but supplies a certain amount of the goods. Under this condition, we will concentrate on the case $p = 2$, i.e. among $m(\geq 2)$ sources there are two factories, each of which can produce the goods and also supply any terminals with them. In Section 2, we reduce the problem to minimization of a univariate function F, each value of which is provided by an ordinary Hitchcock transportation problem with m sources and n terminals. In Section 3, we construct an auxiliary network with $m + n$ nodes associated with the transportation problem giving the values of F. Then we show that local minima of F can be obtained in the course of computing a minimum linear-cost flow in the auxiliary network. Section 4 is devoted to the algorithm for globally minimizing the total cost of the original problem. The number of arithmetic operations required by the algorithm is pseudo-polynomial in the problem input length. Computational results are also reported. In Section 5, we discuss a class of minimum concave-cost flow problems related to our production-transportation problem. We close the paper with some concluding remarks in Section 6.

2. Reduction to Minimization of a Univariate Function

We have two factories, each of which can produce at most a_i units of the goods, $i = 1, 2$, and $m - 2$ warehouses, each with a supply of a_i units, $i = 3, \ldots, m$. The cost of producing y_1 and y_2 units at factories 1 and 2 is given by $g(y_1, y_2)$. We assume that $g : \mathbb{R}^2 \to \mathbb{R}^1$ is a concave function. In [19,20,21], the production cost has not been assumed to be separable, and neither is g throughout this paper. On the other hand, each of n terminals has a demand of b_j units, $j = 1, \ldots, n$. We also know the unit cost c_{ij} of shipping the goods from source i, which is either a factory or a warehouse, to terminal j. Our problem is then formulated as follows:

$$\left|\begin{array}{ll}
\text{minimize} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + g(y_1, y_2) \\
\text{subject to} & \displaystyle\sum_{j=1}^{n} x_{ij} = \begin{cases} y_i, & i = 1, 2 \\ a_i, & i = 3, \ldots, m \end{cases} \\
& \displaystyle\sum_{i=1}^{m} x_{ij} = b_j, \; j = 1, \ldots, n \\
& x_{ij} \geq 0, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n \\
& 0 \leq y_i \leq a_i, \; i = 1, 2,
\end{array}\right. \tag{2.1}$$

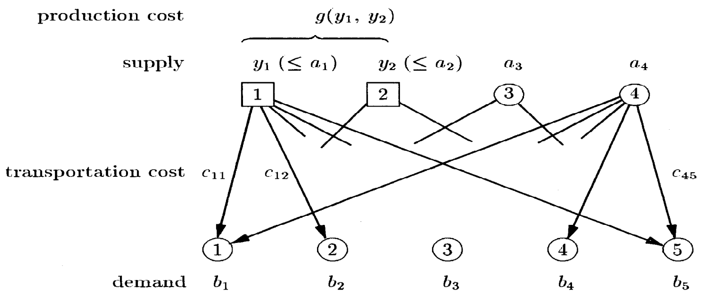

Fig. 2.1. Example of the problem.

where x_{ij}'s, y_1 and y_2 are variables to be determined. We assume that all constants a_i's, b_j's and c_{ij}'s are nonnegative integers. Figure 2.1 shows an example of (2.1) with 4 sources and 5 terminals.

Any feasible solution of (2.1) has to satisfy

$$y_1 + y_2 + \sum_{i=3}^{m} a_i = \sum_{j=1}^{n} b_j. \tag{2.2}$$

Hence, letting

$$\bar{g}(y) = g\left(y, \sum_{j=1}^{n} b_j - \sum_{i=3}^{m} a_i - y\right), \tag{2.3}$$

we can rewrite (2.1) as follows:

$$(\mathrm{TP}) \left| \begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \bar{g}(y) \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} x_{ij} = \begin{cases} y, & i = 1, \\ d - y, & i = 2 \\ a_i, & i = 3, \ldots, m \end{cases} \\ & \displaystyle\sum_{i=1}^{m} x_{ij} = b_j, \ j = 1, \ldots, n \\ & x_{ij} \ge 0, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n \\ & \ell \le y \le u, \quad i = 1, 2, \end{array} \right.$$

where

$$d = \sum_{j=1}^{n} b_j - \sum_{i=3}^{m} a_i, \quad \ell = \max\{0, d - a_2\}, \quad u = \min\{a_1, d\}. \tag{2.4}$$

Note that (TP) can still have multiple locally optimal solutions, since $\bar{g} : \mathbb{R}^1 \to \mathbb{R}^1$ is concave. To exclude trivial cases, we assume in the sequel that $\ell < \mu$. If we fix the value of y in (TP), we have an ordinary Hitchcock transportation problem:

$$
(\mathrm{TP}(y))\quad \left| \begin{array}{ll}
\text{minimize} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{subject to} & \displaystyle\sum_{j=1}^{n} x_{ij} = \begin{cases} y, & i=1 \\ d-y, & i=2 \\ a_i, & i=3,\dots,m \end{cases} \\
& \displaystyle\sum_{i=1}^{m} x_{ij} = b_j,\ j=1,\dots,n \\
& x_{ij} \geq 0, \qquad i=1,\dots,m, \quad j=1,\dots,n.
\end{array}\right.
$$

We can obtain an optimal solution of (TP(y)) in polynomial time if $\ell \leq y \leq u$. We denote it by a vector $\mathbf{x}^*(y)$, whose components are $x^*_{ij}(y)$, $i = 1,\dots,m$, $j = 1,\dots,n$. Let

$$
f(y) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x^*_{ij}(y). \tag{2.5}
$$

Then we see that solving (TP) amounts to finding a global minimum of a function:

$$
F(y) = f(y) + \bar{g}(y). \tag{2.6}
$$

The original problem (2.1) is thus reduced to a minimization problem with a single variable:

$$
\text{(MP) minimize}\{F(y) | \ell \leq y \leq u\},
$$

which we call the *master problem* of (TP).

LEMMA 2.1. *If y^* is a globally optimal solution of (MP), then $(\mathbf{x}^*(y^*), y^*)$ solves (TP), where $\mathbf{x}^*(y^*)$ is an optimal solution of (TP(y^*)).* □

REMARK . The key to the transformation from (TP) to (MP) is a rank-two monotonicity property [18, 23] possessed by the objective function of (TP). Let $\mathbf{z} = (\mathbf{x}, y) \in \mathbb{R}^{mn+1}$ and $h(\mathbf{z}) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \bar{g}(y)$. Then, for $\mathbf{c}^1 = (c_{11}, c_{12}, \dots,$
$c_{m,n-1}, c_{mn}, 0) \in \mathbb{R}^{mn+1}$ and $\mathbf{c}^2 = (0,\dots,0,1) \in \mathbb{R}^{mn+1}$,

$$
\langle \mathbf{c}^1, \mathbf{z} - \mathbf{z}' \rangle \leq 0 \text{ and } \langle \mathbf{c}^2, \mathbf{z} - \mathbf{z}' \rangle = 0 \text{ imply that } h(\mathbf{z}) \leq h(\mathbf{z}'), \tag{2.7}
$$

whenever $\mathbf{z}$ and $\mathbf{z}'$ is feasible to (TP). The function h is certainly concave on $\mathbb{R}^{mn+1}$, but its concavity is embedded in a two-dimensional subspace. In a series of articles [19, 20, 21], Tuy *et al.* have also exploited the property (2.7) to design strongly polynomial algorithms for problems without warehouses. □

3. Characterization of the Univariate Function

Since the objective function F of (MP) is univariate, we can obtain a globally optimal solution y^* by enumerating local minima of $F(y)$ successively from $y = \ell$ to u. To state this systematically, we need to characterize F. As seen in the previous section, F is composed of two functions f and $\bar{g}$. The latter is given, whereas the former requires solving the transportation problem (TP(y)) for all y in the interval $[\ell, u]$. This fact, however, gives the shape of f in outline.

PROPOSITION 3.1. *Function $F : [\ell, u] \to \mathbb{R}^1$ is continuous and piecewise concave.*

Proof. Since (TP(y)) is a parametric right-hand-side linear program, its optimal value function f is piecewise affine, convex and continuous on $[\ell, u]$ (see e.g. [5]). The sum of affine and concave functions is concave [15], so that F is concave on each affine piece of f; moreover, F is continuous on the whole of $[\ell, u]$ due to the continuity of f and $\bar{g}$. □

We immediately see from the proposition that among extreme points of affine pieces of f exists a global minimizer y^* of F. This kind of piecewise concave analysis has its origins in the work by Zangwill in the 60s [24, 245]. He assumed a piecewise concave inventory cost in a dynamic lot sizing problem.

Let us suppose $f(y')$ is given for an arbitrary $y' \in [\ell, u)$, and hence an optimal solution $\mathbf{x}^*(\mathbf{y}')$ of (TP(y')) is available. In the rest of the section, we will develop a procedure to compute $f(y' + \delta)$ for sufficiently small $\delta \geq 0$. Using this procedure, we will specify the affine piece of f containing y'.

3.1. Minimum cost flow in an auxiliary network

We first construct an auxiliary graph $G(y') = (M, N, A(y'))$ associated with (TP(y')), where $M = \{1, \ldots, m\}$ and $N = \{1, \ldots, n\}$ are the sets of sources and terminals, respectively, of (TP) and $A(y')$ is a set of directed arcs. Based on the optimal solution $\boldsymbol{x}^*(\mathbf{y}')$ of (TP(**y'**)), we define $\mathbf{A}(\mathbf{y'})$ and a capacity $u_{ij}(y')$ of each arc $(i, j) \in A(y')$ as follows (see also Figure 3.1): For each pair (i, j) such that $i \in M$ and $j \in M$ and $j \in N$, let

$$(i, j) \in A(y'), \quad u_{ij}(y') = +\infty, \tag{3.1}$$

$$(j, i) \in A(y'), \quad u_{ji}(y') = x^*_{ij}(y') \text{ if } x^*_{ij}(y') > 0. \tag{3.2}$$

In addition, for each $(i, j) \in A(y')$ we define a cost:

$$c_{ij}(y') = \begin{cases} c_{ij} & \text{if } i \in M, j \in N \\ -c_{ij} & \text{if } j \in M, i \in N. \end{cases} \tag{3.3}$$

In Figure 3.1, the right arcs are constructed from the left one. Let $\mathbf{c}(y')$ and $\mathbf{u}(y')$ denote the vectors of $c_{ij}(y')$'s and $u_{ij}(y')$'s, respectively. Then we have the following problem in network $\mathcal{N}(y') = (G(y'), \mathbf{c}(y'), \mathbf{u}(y'))$:

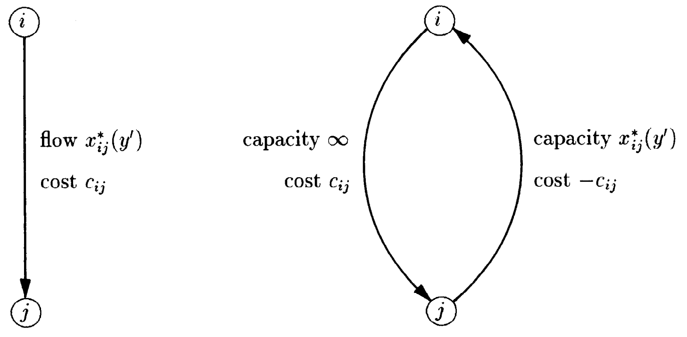

Fig. 3.1. Arcs in the auxiliary network $\mathcal{N}(y')$.

$$(\mathrm{P}(\delta;y'))\quad \left|\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(i,j)\in A(y')} c_{ij}(y')z_{ij} \\ \text{subject to} & \displaystyle\sum_{j\in V(i)} z_{ji} - \sum_{j\in W(i)} z_{ji} = \begin{cases} \delta, & i=1 \\ -\delta, & i=2 \\ 0, & i\in M\cup N\backslash\{1,2\} \end{cases} \\ & 0\le z_{ij}\le u_{ij}(y'), \qquad\qquad (i,j)\in A(y'), \end{array}\right.$$

where z_{ij}'s are variables, $V(i)=\{j\in M\cup N|(i,j)\in A(y')\}$ and $W(i)=\{j\in M\cup N|(j,i)\in A(y')\}$. Since (P($\delta;y'$)) is a minimum linear-cost flow problem in $\mathcal{N}(y')$ with a single source $s=1\in M$ and a single sink $t=2\in M$, we can solve it efficiently using available algorithms. Let us denote an optimal solution of (P($\delta;y'$)) by $\mathbf{z}^*(\delta;y')$, whose components are $z^*_{ij}(\delta;y')$, $(i,j)\in A(y')$.

LEMMA 3.2. *For each (i,j) such that $i\in M$ and $j\in N$, let*

$$x^*_{ij}(\delta;y') = \begin{cases} z^*_{ij}(\delta;y') & \text{if } x^*_{ij}(y')=0 \\ x^*_{ij}(y')+z^*_{ij}(\delta;y')-z^*_{ji}(\delta;y') & \text{otherwise.} \end{cases} \tag{3.4}$$

Then $\mathbf{x}^*(\delta;y')$, whose components are $x^*_{ij}(\delta;y')$'s, is optimal to (TP($y'+\delta$)).

Proof. For an arbitrary feasible solution $\mathbf{z}$ of (P($\delta;y'$)), let

$$x'_{ij} = \begin{cases} z_{ij} & \text{if } x^*_{ij}(y')=0 \\ x^*_{ij}(y')+z_{ij}-z_{ji} & \text{otherwise.} \end{cases}$$

Then we have

$$\begin{aligned} \sum_{j=1}^{n} x'_{ij} &= \sum_{j=1}^{n} x^*_{ij}(y') + \left(\sum_{j\in V(i)} z_{ij} - \sum_{j\in W(i)} z_{ji}\right) \\ &= \begin{cases} y'+\delta, & i=1 \\ d-y'-\delta, & i=2 \\ a_i, & i=3,\ldots,m, \end{cases} \\ \sum_{i=1}^{m} x'_{ij} &= \sum_{i=1}^{m} x^*_{ij}(y') - \left(\sum_{i\in V(j)} z_{ji} - \sum_{i\in W(j)} z_{ij}\right) = b_j, j=1,\ldots,n, \end{aligned}$$

and

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x'_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x^*_{ij}(y') + \sum_{(i,j)\in A(y')} c_{ij}(y')z_{ij}.$$

Hence, the feasible set of (P($\delta; y'$)) represents possible adjustments of $\mathbf{x}^*(y')$ to a slight change in y'. Among such adjustments $\mathbf{z}^*(\delta; y')$ requires the least cost, which apparently implies that $\mathbf{x}^*(\delta; y')$ defined by (3.4) is optimal to (TP($y' + \delta$)). □

3.2. Application of the primal-dual algorithm

We next try applying the primal-dual algorithm [3] to the auxiliary problem (P($\delta; y'$)). The algorithm begins with a zero flow in $\mathcal{N}(y')$, and augments it by adding some flow along a directed path from source s to sink t with the least cost in $\mathcal{N}(y')$. To find such an augmenting path, we need to solve a shortest path problem in $G(y')$ with the arc length $\mathbf{c}(y')$. It follows from (3.2) that there exists some $j \in N$ such that $(j, 2) \in A(y')$ as long as $y' < d$. Hence, we have a shortest path $\pi(y') \subset A(y')$ from s to $t(= 2 \in M)$. Let

$$\bar{\delta} = \min\{u_{ij} | (i,j) \in \pi(y')\}. \tag{3.5}$$

LEMMA 3.3. *If* $0 \le \delta \le \bar{\delta}$, *then*

$$z^*_{ij}(\delta; y') = \begin{cases} \delta & \textit{if } (i,j) \in \pi(y') \\ 0 & \textit{otherwise} \end{cases} \tag{3.6}$$

is an optimal solution of (P($\delta; y'$)).

Proof. Follows from a well-known result on the primal-dual algorithm for minimum cost flow problems (see e.g. [11]). □

It follows from (3.4) and (3.6) that

$$\begin{aligned} f(y'+\delta) &= \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x^*_{ij}(\delta; y') \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x^*_{ij}(y') + \sum_{(i,j)\in A(y')} c_{ij}(y')z^*_{ij}(\delta; y') \\ &= f(y') + \delta \sum_{i,j)\in \pi(y')} c_{ij}(y'). \end{aligned} \tag{3.7}$$

This implies that f is an affine function over the interval $[y', y' + \bar{\delta}]$.

4. Solution Method for the Problem

The above observation about the univariate function F leads to the following solution method for (MP): Let $y_0 = \ell$ and let (P($\delta; y_0$)) be the auxiliary problem of

(TP(y_0)). From Lemmas 3.2 and 3.3, we can have an interval $[y_0, y_0 + \bar{\delta}]$, where f is affine, in the first step of solving (P($\delta; y_0$)) by the primal-dual algorithm. In the same way, we construct (P($\delta; y_1$)) for $y_1 = y_0 + \bar{\delta}$ and determine another interval $[y_1, y_2]$. Repeating this process, we generate a sequence of intervals $[y_0(= \ell), y_1]$, $[y_1, y_2], \ldots, [y_{q-1}, y_q(= u)]$ such that f is affine on each $[y_{k-1}, y_k]$, $k = 1, \ldots, \delta$. Since F is concave on $[y_{k-1}, y_k]$, its minimum over the interval is attained at either y_{k-1} or y_k. Therefore, a globally optimal solution of the master problem (MP) is given by

$$y^* \in \text{argmin}\{F(y) | y = y_0, y_1, \ldots, y_q\}. \tag{4.1}$$

In practice, we need not solve each (P($\delta; y_k$)) from scratch. Recall that the primal-dual algorithm [3] builds up a flow step by step, by adding flows along augmenting paths in some auxiliary network $\mathcal{N}'$. At each iteration, we augment the flow along a least-cost augmenting path π' in $\mathcal{N}'$ until the flow reaches the capacity of π', and then update the auxiliary network $\mathcal{N}'$. When we apply this procedure entirely to (P($u - \ell; \ell$)), the auxiliary network $\mathcal{N}'$ and the augmenting path π' at the kth iteration just correspond to $\mathcal{N}(y_{k-1})$ and $\pi(y_{k-1})$, respectively, and the capacity of π' is given by $\min\{u_{ij} | (i,j) \in \pi(y_{k-1})\}$.

4.1. Algorithm for the original problem

The algorithm for solving the original problem (TP) is summarized into the following:

Algorithm PDM.

Phase I. Solve a transportation problem (TP(ℓ)) and let $\mathbf{x}^*(\ell)$ be an optimal solution. Let $f(\ell) = \sum_{i=1}^{m} \sum_{i=1}^{n} c_{ij} x_{ij}^*(\ell)$. Initialize the incumbent:

$$\mathbf{x}^* = \mathbf{x}^*(\ell), \quad y^* = \ell, \quad F^* = f(\ell) + \bar{g}(\ell).$$

Phase II. Construct the auxiliary network $\mathcal{N}(\ell) = (G(\ell) = (M, N, A(\ell)), \mathbf{c}(\ell), \mathbf{u}(\ell))$ of (TP(ℓ)) according to (3.1)–(3.3). Let $s = 1 \in M$ and $t = 2 \in M$. Begin the following with $y_0 = \ell$ and $k = 0$:

1° Compute a shortest path $\pi(y_k)$ from s to t in $G(y_k)$ with the arc length $\mathbf{c}(y_k)$. Let $\bar{\delta} = \min\{u_{ij} | (i,j) \in \pi(y_k)\}$. If $y_k + \bar{\delta} > u$, then let $\bar{\delta} = u - y_k$.

2° Let $y_{k+1} = y_k + \bar{\delta}$. For each (i,j) such that $i \in M$ and $j \in N$, let

$$x_{ij}^*(y_{k+1}) = \begin{cases} x_{ij}^*(y_k) + \bar{\delta} & \text{if } (i,j) \in \pi(y_k) \\ x_{ij}^*(y_k) - \bar{\delta} & \text{if } (j,i) \in \pi(y_k) \\ x_{ij}^*(y_k) & \text{otherwise.} \end{cases}$$

Also let $f(y_{k+1}) = f(y_k) + \bar{\delta} \sum_{(i,j) \in \pi(y_k)} c_{ij}(y_k)$.

3° If $f(y_{k+1}) + \bar{g}(y_{k+1}) < F^*$, then revise the incumbent:

$$\mathbf{x}^* = \mathbf{x}^*(y_{k+1}), \quad y^* = y_{k+1}, \quad F^* = f(y_{k+1}) + \bar{g}(y_{k+1}).$$

4° If $y_{k+1} = u$, then terminate. Otherwise, update the auxiliary network $\mathcal{N}(y_k)$ according to (3.1)–(3.3) and let $\mathcal{N}(y_{k+1})$ be the resultant network. Let $k = k + 1$ and return to 1°. □

Note that Phase II of this algorithm is nothing but the primal-dual algorithm for solving the minimum cost flow problem (P($u - \ell; \ell$)) if we drop step 3°, in which a globally optimal solution $(\mathbf{x}^*, y^*)$ of (TP) is computed.

Let us denote by $S(m, n)$ the running time needed to solve a shortest path problem with mn arcs and $m + n$ nodes, and by $T(m, n)$ that to solve a Hitchcock transportation problem with m sources and n terminals. As is well known (see e.g. [1]), both $S(m, n)$ and $T(m, n)$ are lower-order polynomial functions of m and n.

THEOREM 4.1. *Algorithm PDM yields a globally optimal solution $(\mathbf{x}^*, y^*)$ of (TP) in $O(CS(m, n) + T(m, n))$ arithmetic operations and $O(C)$ evaluations of g, where $C = u - \ell$.*

Proof. Phase I requires $T(m, n)$ arithmetic operations to solve a transportation problem (TP(ℓ)). Phase II computes a shortest path $\pi(y_k)$ and the value $\bar{g}(y_{k+1})$ at each iteration. The total number of iterations is bounded by $C = u - \ell$, since $\bar{\delta} \geq 1$ on the assumption that all constants are integral in (TP). Hence, the assertion follows. □

REMARKS. 1) In step 1° of Phase II, we cannot use Dijkstra's algorithm directly to compute the shortest path $\pi(y_k)$ because some components of $\mathbf{c}(y_k)$ are negative. However, on the assumption that all c_{ij}'s are nonnegative in (TP), we can transform $\mathbf{c}(y_k)$ into a nonnegative vector by introducing node potentials. Then $\pi(y_k)$ can be computed in time $S(m, n) = O(mn + (m + n) \log(m + n))$ [4]. The readers are referred to any textbook on network flows, e.g. [1] for further details.

2) The production cost $\bar{g}$ is often assumed to be piecewise concave but discontinuous, e.g. a fixed-charge cost function. Algorithm PDM can still handle discontinuous $\bar{g}$'s as long as they are lower semi-continuous, with a minor modification. Let us divide each $[y_{k-1}, y_k]$ at discontinuous points of $\bar{g}$. Then $[\ell, u]$ is partitioned into r $(\geq q)$ subintervals $[\eta_{k'-1}, \eta_{k'}]$, $k' = 1, \ldots, r$, where $\eta_{k'}$ is either a y_k or a discontinuous point of $\bar{g}$. Since $F = f + \bar{g}$ is concave on the interior of each $[\eta_{k'-1}, \eta_{k'}]$, it achieves the minimum on $[\ell, u]$ at some $\eta_{k'}$ by the lower semi-continuity. Hence, to locate y^* in $[\ell, u]$, we need only to compute the values of F at discontinuous points of $\bar{g}$ as well as y_k's. □

4.2. Numerical example

Let us illustrate algorithm PDM using a simple instance of (TP) given by the table below:

source\terminal	t_1	t_2	t_3	t_4	supply	capacity
s_1	12	1	3	4	y	200
s_2	4	9	6	2	$300 - y$	200
s_3	2	6	2	10	150	—
demand	80	180	120	70	450	—

where each entry (s_i, t_j) represents the transportation cost c_{ij}. The production cost of factories s_1 and s_2 is assumed to be

$$\bar{g}(y) = 100.0 \cdot \sqrt{y}.$$

The lower and the upper bounds of y are respectively

$$\ell = 100, \quad u = 200.$$

In Phase I, we solve a transportation problem (TP(100)). Then an optimal solution $\mathbf{x}^*(100)$ is as follows:

	t_1	t_2	t_3	t_4	supply
s_1	0	100	0	0	100
s_2	80	50	0	70	200
s_3	0	30	120	0	150

We also initialize the incumbent:

$$\begin{aligned} \mathbf{x}^* &= \mathbf{x}^*(100), \quad y^* = 100, \\ F^* &= f(100) + \bar{g}(100) = 1430 + 1000.00 = 2430.00. \end{aligned}$$

In Phase II, for each arc (i, j) with $x^*_{ij}(100) > 0$ we put a reverse arc (j, i) with capacity $x^*_{ij}(100)$ and cost $-c_{ij}$, i.e.

arc(t_2, s_1) with capacity 100 and cost -1
arc(t_1, s_2) with capacity 80 and cost -4
arc(t_2, s_2) with capacity 50 and cost -9
arc(t_4, s_2) with capacity 70 and cost -2
arc(t_2, s_3) with capacity 30 and cost -6
arc(t_3, s_3) with capacity 120 and cost -2,

and denote by $\mathcal{N}(100)$ the resultant network. Letting $y_0 = 100$, we proceed to the repeating process.

At the first iteration, we compute a shortest path $\pi(100)$ from $s = \mathrm{s}_1$ to $t = \mathrm{s}_2$ in $\mathcal{N}(100)$ and obtain:

$$\pi(100) = (\mathrm{s}_1, \mathrm{t}_2, \mathrm{s}_2), \quad \bar{\delta} = \min\{\infty, 50\} = 50.$$

Then we let $y_1 = 150$ and compute $\mathbf{x}^*(150)$, which is given by

	t_1	t_2	t_3	t_4	supply
s_1	0	150	0	0	150
s_2	80	0	0	70	150
s_3	0	30	120	0	150

Since

$$f(150) + \bar{g}(150) = 1030 + 1224.75 = 2254.75 < F^*,$$

we revise the incumbent as follows:

$$\mathbf{x}^* = \mathbf{x}^*(150), y^* = 150, F^* = f(150) + \bar{g}(150) = 2254.75.$$

According to the same rule as before, we update $\mathcal{N}(100)$ based on $\mathbf{x}^*(150)$, and denote by $\mathcal{N}(150)$ the resultant network.

At the second iteration, we compute a shortest path in $\mathcal{N}(150)$:

$$\pi(150) = (\mathrm{s}_1, \mathrm{t}_2, \mathrm{s}_3, \mathrm{t}_1, \mathrm{s}_2), \bar{\delta} = \{\infty, 30, \infty, 80\} = 30.$$

We let $y_2 = 180$ and compute $\mathbf{x}^*(180)$:

	t_1	t_2	t_3	t_4	supply
s_1	0	180	0	0	180
s_2	50	0	0	70	120
s_3	30	0	120	0	150

Since

$$f(180) + \bar{g}(180) = 820 + 1341.64 = 2161.64,$$

we revise the incumbent again:

$$\mathbf{x}^* = \mathbf{x}^*(180), y^* = 180, F^* = f(180) + \bar{g}(180) = 2161.64.$$

We also update $\mathcal{N}(150)$ based on $\mathbf{x}^*(180)$ and obtain $\mathcal{N}(180)$.

At the third iteration, we compute a shortest path in $\mathcal{N}(180)$:

$$\pi(180) = (\mathrm{s}_1, \mathrm{t}_3, \mathrm{s}_3, \mathrm{t}_1, \mathrm{s}_2), \bar{\delta} = \min\{\infty, 120, \infty, 50\} = 50.$$

Since $y_2 + \bar{\delta} = 230 > u = 200$, we modify $\bar{\delta} = 20$. We let $y_3 = 200$ and compute $\mathbf{x}^*(200)$:

	t_1	t_2	t_3	t_4	supply
s_1	0	180	20	0	200
s_2	30	0	0	70	100
s_3	50	0	100	0	150

Since

$$f(200) + \bar{g}(200) = 800 + 1414.21 = 2214.21 > F^*$$

and $y_3 = u$, we find that the current $(\mathbf{x}^*, y^*)$ is a globally optimal solution of our instance.

4.3. COMPUTATIONAL RESULTS

Before concluding this section, we will report computational results of testing PDM on randomly generated instances of (TP).

Algorithm PDM was coded in C language and run on a microSPARC II (70 MHz). The procedure employed to solve (TP(ℓ)) in Phase I was the primal-dual algorithm [3], which was basically the same as Phase II. Shortest paths in step 1° were computed by Dijkstra's common algorithm. The running time of our computer code is therefore bounded by $O((B+C)(m+n)^2)$, where $b = \sum_{j=1}^{n} b_j$. The data values of b_j's and c_{ij}'s were drawn from a uniform distribution of integers in the range [0, 20]. The production function was defined as $\bar{g}(y) = 100.0\sqrt{y}$.

Table 4.1 shows the average performance of PDM over ten instances for each (m, n), which was varied from (20, 60) up to (50, 100). The values of a_i's were set as follows:

$$a_i = \left\lfloor \sum_{j=1}^{n} b_j \middle/ (m-1) \right\rfloor, i = 3, \ldots, m; a_1 = a_2 = \sum_{j=1}^{n} b_j - \sum_{i=3}^{m} a_i.$$

Therefore, C was equal to $a_1 = a_2$. The row labeled 'Phase I' gives the average CPU time in seconds and the average number of iterations required in Phase I; the row labeled 'Phase II' gives those required in Phase II; the row labeled 'Total' gives the totals of those numbers.

Table 4.1. Computational results of PDM for random C.

	m	20	20	20	30	30	30
	n	60	80	100	60	80	100
Phase I:	CPU time	11.81	21.98	45.78	14.68	32.95	61.48
	iterations	94.9	114	143	103	132	154
Phase II:	CPU time	1.798	2.777	6.097	1.710	4.103	6.358
	iterations	13.8	15.0	19.8	12.5	17.1	16.4
Total:	CPU time	13.61	24.76	51.88	16.39	37.05	67.84
	iterations	109	129	163	116	149	170
	m	40	40	40	50	50	50
	n	60	80	100	60	80	100
Phase I:	CPU time	22.35	45.70	83.51	32.02	61.99	108.8
	iterations	116	142	168	128	154	179
Phase II:	CPU time	2.720	4.570	9.641	4.585	7.955	10.78
	iterations	14.6	14.7	19.9	19.0	20.4	20.2
Total:	CPU time	25.07	50.27	93.15	36.61	69.95	119.6
	iterations	131	157	188	147	175	199

Table 4.2. Computational results of PDM when $(m, n) = (30, 80)$.

	C	10	50	100	150	200	250
Phase I:	CPU time	40.07	40.25	41.88	43.74	45.16	46.99
	iterations	132	133	138	145	149	155
Phase II:	CPU time	1.278	5.095	9.353	14.06	16.35	20.55
	iterations	4.40	18.5	31.0	48.2	56.0	70.3
Total:	CPU time	41.35	45.35	51.23	57.79	61.51	67.54
	iterations	137	152	169	193	205	225

Table 4.2 shows the average performance of PDM against the variation of C when (m, n) was fixed at (30,80). Both the values of a_1 and a_2 were set equal to C, and those of a_i, $i = 3, \dots, m$, were set as follows:

$$a_i = \left\lfloor \left(\sum_{j=1}^{n} b_j - C \right) \Big/ (m-2) \right\rfloor, i = 4, \dots, m;$$

$$a_3 = \left(\sum_{j=1}^{n} b_j - C \right) - \sum_{i=4}^{m} a_i .$$

For each value of C, varied from 10 to 250, the same statistics as Table 4.1 are listed in Table 4.2.

We see from these two tables that Phase II requires much less CPU time and iterations than Phase I for all test problems. This relation might be reversed if we use a polynomial algorithm in Phase I. In that case, however, the total computational time will naturally be improved at the same time.

5. A Minimum Concave-Cost Flow Problem

In both combinatorial and global optimization, one of the most attractive but most difficult problems is the minimum concave-cost flow problem. To solve this NP-hard problem, many algorithms have been developed so far, and some of them turned out to be promising for some special cases (see [7,8] and references therein). Especially when both the numbers of sources and nonlinear-cost arcs are fixed, uncapacitated problems can be solved in polynomial time [9, 12, 19, 22]. In this section, we will show that capacitated problems with a single nonlinear-cost arc can be transformed into the class (TP) of production-transportation problems and hence solved by algorithm PDM in pseudo-polynomial time.

Let $G = (N, A)$ be a directed graph consisting of a set N of nodes and a set A of directed arcs. We associate with each arc $(i, j) \in A$ a concave cost $g_{ij} : \mathbb{R}^1 \to \mathbb{R}^1$ and a capacity $u_{ij} \geq 0$, and with each node $i \in N$ a number b_i, which indicates its supply or demand depending on whether $b_i > 0$ or $b_i < 0$. Then the minimum concave-cost flow problem is formulated as follows:

$$(\text{FP}) \quad \left| \begin{array}{ll} \text{minimize} & \displaystyle\sum_{(i,j)\in A} g_{ij}(x_{ij}) \\ \text{subject to} & \displaystyle\sum_{j\in V(i)} x_{ji} - \sum_{j\in W(i)} x_{ji} = b_i, i \in N \\ & 0 \leq x_{ij} \leq u_{ij}, \qquad (i,j) \in A, \end{array} \right.$$

where x_{ij}'s are variables, $V(i) = \{j \in N | (i,j) \in A\}$ and $W(i) = \{j \in N | (j,i) \in A\}$. We assume that all constants are integral, and for simplicity that (FP) has a feasible flow. In this problem, we are concerned with the case where all g_{ij}'s except one, say g_{vw}, are linear functions, i.e. for some nonnegative integers c_{ij}'s,

$$g_{ij}(x_{ij}) = c_{ij}x_{ij}, (i,j) \in A \backslash \{(v,w)\}. \tag{5.1}$$

Given such an instance of (FP), we will construct an instance of (TP).

If flow x_{vw} of the nonlinear-cost arc (v, w) is fixed at an arbitrary value y, we have a minimum linear-cost flow problem:

$$(\text{FP}(y)) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{(i,j)\in A'} c_{ij}x_{ij} & \\ \text{subject to} & \displaystyle\sum_{j\in V'(i)} x_{ij} - \sum_{j\in W'(i)} x_{ji} = \left\{ \begin{array}{ll} b_v - y, & i = v \\ b_w + y, & i = w \\ b_i, & i \in N\backslash\{v,w\} \end{array} \right. \\ & 0 \leq x_{ij} \leq u_{ij}, & (i,j) \in A', \end{array} \right.$$

where $A' = A\backslash\{(v,w)\}$, $V'(i) = \{j \in N | (i,j) \in A'\}$ and $W'(i) = \{j \in N | (j,i) \in A'\}$. As is well known, we can transform (FP(y)) into a Hitchcock

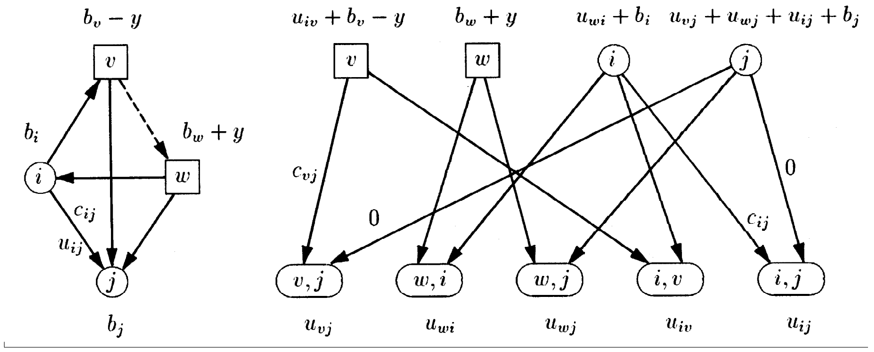

Fig. 5.1. Transformation from (FP(y)) to (FP$'$(y)).

transportation problem in the following manner (see e.g. [1] for details): Let us regard N as the set of sources and A' as the set of terminals. For each $(i, j) \in A'$ we first define two directed arcs $(i, (i, j))$ and $(j, (i, j))$, and assign cost c_{ij} to the former and cost zero to the latter. We next let $\sum_{j \in W'(i)} u_{ji} + b'_i$ be the supply of source $i \in N$ and u_{ij} be the demand of terminal $(i, j) \in A'$, where $b'_v = b_v - y$, $b'_w = b_w + y$ and $b'_i = b_i$ for each $i \in N \backslash \{v, w\}$. Figure 5.1 shows the transformation, where the right network is transformed from the left one.

Now we have the following problem equivalent to (FP(y)):

$$(\mathrm{FP}'(y)) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{(i,j)\in A'} c_{ij}\xi_{i(i,j)} & \\ \text{subject to} & \displaystyle\sum_{(i,j)\in A'} \xi_{i(i,j)} + \sum_{(j,i)\in A'} \xi_{i(j,i)} = \left\{ \begin{array}{l} a'_v - y,\ i = v \\ a'_w + y,\ i = w \\ a'_i,\ i \in N\backslash\{v, w\} \end{array} \right. & \\ & \xi_{i(i,j)} + \xi_{j(i,j)} = u_{ij}, & (i,j) \in A' \\ & \xi_{i(j,k)} \geq 0, & i \in N, (j,k) \in A', \end{array} \right.$$

where $\xi_{i(j,k)}$'s are variables and

$$a'_i = \sum_{j \in W'(i)} u_{ji} + b_i, \quad i \in N. \tag{5.2}$$

It is easy to see that our instance of (FP) can be solved if we minimize the sum of the optimal value of (FP$'$(y)) and $g_{vw}(y)$ subject to $0 \leq y \leq u_{vw}$. In other words, a globally optimal solution of (FP) with a single nonlinear-cost arc can be obtained if we solve a production-transportation problem:

$$\left|\begin{array}{lll}\text{minimize} & \displaystyle\sum_{(i,j)\in A'} c_{ij}\xi_{i(i,j)} + g_{vw}(y) & \\ \text{subject to} & \displaystyle\sum_{(i,j)\in A'} \xi_{i(i,j)} + \sum_{(j,i)\in A'} \xi_{i(j,i)} = \begin{cases} a'_v - y, & i = v \\ a'_w + y, & i = w \\ a'_i, & i \in N\backslash\{v,w\} \end{cases} & \\ & \xi_{i(i,j)} + \xi_{j(i,j)} = u_{ij}, & (i,j) \in A' \\ & \xi_{i(j,k)} \geq 0, & i \in N, (j,k) \in A', \\ & 0 \leq y \leq u_{vw}, & \end{array}\right.$$

which apparently belongs to (TP).

6. Concluding Remarks

The converse of the result in the previous section is also possible. Let us introduce an artificial source v with a supply of d units and two artificial arcs $(v,1)$ and $(v,2)$ in (TP). We define the capacity u_{vi} and cost g_{vi} of arc (v,i) as follows:

$$u_{vi} = a_i, \quad i = 1,2; \quad g_{v1}(x_{v1}) = \bar{g}(x_{v1}), \quad g_{v2}(x_{v2}) = 0.$$

We also let $b_1 = b_2 = 0$, i.e. factories 1, 2 are now intermediate nodes. the resultant problem is then a capacitated minimum concave-cost flow problem with a single nonlinear-cost arc $(v,1)$, which can be solved by some algorithms, e.g. developed by Erickson *et al.* [2] and by Klinz and Tuy [12]. While the running time of the former based on dynamic programming is exponential in the number n of terminals, the latter runs in pseudo-polynomial time as our algorithm does.

Klinz and Tuy's algorithm (abbr. KT) is closely related to algorithm PDM. Both algorithms are based on a parametric approach to rank-two quasiconcave minimization developed by Tuy *et al.* [18,23]. Moreover, they have a duality relationship: PDM solves the parametric *right-hand-side* linear program (TP(y)) and KT solves a parametric *cost* linear program, to find a globally optimal solution. The optimal value function of the parametric cost linear program is piecewise affine but concave. For each affine piece, Klinz and Tuy proposed to solve a minimum linear-cost flow problem. Consequently, KT requires $O((Dm'\log n')(m' + n'\log n'))$ arithmetic operations, where $m' = mn$, $n' = m + n$ and $D = \min\{a_1, \sum_{j=1}^{n} b_j\}$. On the other hand, if we use Orlin's algorithm [16] in Phase I, the number of arithmetic operations required by PDM is bounded by $O((C + m'\log n')(m' + n'\log n'))$, which is rather improved upon algorithm KT. The main reason for this difference is that we had only to solve a shortest path problem for each break point of the piecewise affine function f.

Since the core of the parametric approach is to specify the optimal value function of a parametric linear program, we could devise some other procedures for doing so. For example, we can specify f by applying the parametric network simplex algorithm to (TP(y)) as well (see e.g. Chapter 11 of [1]). Nevertheless, algorithm PDM proposed in this paper possesses at least two advantages: (1) the computer code is easy to write because Phases I and II can be implemented with a common existing code: (2) for certain concave production costs, the algorithm is a fully polynomial approximation scheme [6], i.e. it yields a globally ϵ-optimal solution in polynomial time. The second matter will be discussed in detail in the subsequent article [14].

Acknowledgement

The authors are grateful to the editor and two anonymous reviewers for their valuable comments and suggestions, which have considerably improved the earlier version of this paper. One of the reviewers drew our attention to an application of the parametric network simplex algorithm mentioned in Section 6.

References

1. Ahuja, R.K., T.L. Magnanti and J.B. Orlin, *Network flows: Theory, Algorithms and Applications*, Prentice Hall, N.J., (1993).
2. Erickson, R.E., C.L. Monma and A.F. Veinott, “Send-and-split method for minimum-concave-cost network flows”, *Mathematics of Operations Research* **12** (1987), 634–664.
3. Ford, L.R. and D.R. Fulkerson, *Flows in Networks*, Princeton University Press, N.J., (1962).
4. Fredman, M.L. and R.E. Tarjan, “Fibonacci heaps and their uses in improved network optimization algorithms”, *Journal of ACM* **34** (1987), 596–615.
5. Gal, T., “Linear parametric programming — a brief survey”, *Mathematical Programming Study* **21** (1984), 43–68.
6. Garey, M.S. and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, N.Y., (1979).
7. Guisewite, G.M., “Network problems”, in R. Horst and P.M. Pardalos (eds.), *Handbook of Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1995).
8. Guisewite, G.M. and P.M. Pardalos, “Minimum concave-cost network flow problems: applications, complexity and algorithms”, *Annals of Operations Research* **25** (1990), 75–100.
9. Guisewite, G.M. and P.M. Pardalos, “A polynomial time solvable concave network flow problem”, *Networks* **23** (1993), 143–149.
10. Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer-Verlag, Berlin, (1990).
11. Iri, M., “A new method of solving transportation network problems”, *Journal of the Operations Research Society of Japan* **3** (1960), 27–87.
12. Klinz, B. and H. Tuy, “Minimum concave-cost network flow problems with a single nonlinear arc cost”, Dungzhu Du and P.M. Pardalos (eds.), *Network Optimization Problems*, World Scientific, Singapore, (1993), 125–143.
13. Kuno, T. and T. Utsunomiya, “A decomposition algorithm for solving certain classes of production-transportation problems with concave production cost”, *Journal of Global optimization* **8** (1996), 67–80.
14. Kuno, T. and T. Utsunomiya, “Minimizing a linear multiplicative-type function under network flow constraints”, Technical Report ISE-TR-95-124, Institute of Information Sciences and Electronics, University of Tsukuba, Ibaraki, (1995), to appear in *Operations Research Letters*.
15. Mangasarian, O.L., *Nonlinear Programming*, McGraw-Hill (N.Y., 1969).
16. Orlin, J.B., “A faster strongly polynomial minimum cost flow algorithm”, *20th ACM Symposium on Theory of Computing* (1988), 377–387.

17. Pardalos, P.M. and S.A. Vavasis, “Quadratic programming with one negative eigenvalue is NP-hard”, *Journal of Global Optimization* **1** (1991), 15–22.
18. Tuy, H., “The complementary convex structure in global optimization”, *Journal of Global Optimization* **2** (1992), 21–40.
19. Tuy, H., N.D. Dan and S. Ghannadan, “Strongly polynomial time algorithms for certain concave minimization problems on networks”, *Operations Research Letters* **14** (1993), 99–109.
20. Tuy, H., S. Ghannadan, A. Migdalas and P. Väbrand, “Strongly polynomial algorithm for a production-transportation problem with concave production cost”, *Optimization* **27** (1993), 205–227.
21. Tuy, H., S. Ghannadan, A. Migdalas and P. Värbrand, “Strongly polynomial algorithm for a production-transportation problem with a fixed number of nonlinear variables”, Preprint, Department of Mathematics, Linköping, (1993) to appear in *Mathematical Programming*.
22. Tuy, H., S. Ghannadan, A. Migdalas and P. Värbrand, “The minimum concave cost network flow problems with fixed number of sources and nonlinear arc costs”, *Journal of Global Optimization* **6** (1995), 135–151.
23. Tuy, H. and B.T. Tam, “An efficient solution method for rank two quasiconcave minimization problems”, *Optimization* **24** (1992), 43–56.
24. Zangwill, W.L., “A deterministic multi-product, multi-facility production and inventory system”, *Operations Research* **14** (1966), 486–508.
25. Zangwill, W.L., “A backlogging model and multi-echelon model of a dynamic economic lot size production system — a network approach”, *Management Science* **15** (1969), 506–527.